Using the Software AG Editor

This section describes the functions that are available for all object types using main commands. Note that the commands available to you depend on the application you are using.

This section covers the following topics:

- Change a Specific Character String
- Center Data
- Copy Lines
- Copy a Window with Data
- Define Horizontal and Vertical Boundaries
- Define a Mask Line
- Delete Lines
- Display Boundary, Tab and Column Positions
- Exclude/Include Lines from Display
- Find a Specific Character String
- Include Lines from Display
- Insert Lines
- Justify Data
- Leave the Editor
- Locate a Specific Line
- Move Lines
- Order Data Between Specified Boundaries
- Overlay Lines
- Repeat Lines
- Scroll Data on the Editor Screen
- Sort Lines in Alphabetical Order
- Store Data
- Use the Physical/Logical Tabulator

Change a Specific Character String

You can find and automatically change a given character string into another character string using the CHANGE main command.

The same operands as for the FIND command can be used with the CHANGE command. For the CHANGE command, however, the ALL directions operand means change **all occurrences of the specified string**.

After the change is performed, the message **==chg>** appears in the line command field of the changed line.

Examples of CHANGE Command

Command	Explanation
CHG ’LOW’ ’HIGH’	Change the first occurrence **low** into **high** (upper and lower case ignored).
CHG C’ops’ ’SPF’ .X .Y 28 32 all	Change **ops** (exactly as entered here) into **SPF***;* change all occurrences in the block labelled ***.X*** and **.Y** and between the Columns 28 and 32.
Repeated CHANGE commands:	
CHG * ’NEW’	Change the next occurrence of the string specified in the last CHANGE command to new string **NEW**.
CHG ’OLD’ *	Change string **OLD** into the same new string as specified in the last CHANGE command.

If single quotation marks are part of the string to be changed, you must use a different separator in the CHANGE command, for example double quotation marks:
CHANGE "’*string1*’" "’*string2*’".

The screens below illustrate the second example.

To change all occurrences of the string *ops* to the string *SPF* between lines 140 and 170 and within columns 28 to 32 issue the following command:

```
CHG C’ops’ ’SPF’ .X .Y 28 32 ALL
```

The screen appears as shown below before the command is issued. The command appears in the COMMAND line on this screen:

```
S*>>EDIT-NAT:NSPF101(JOB1JCL)-Program->Report-Free-29K--------- columns 001 072
   COMMAND===> CHG C’OPS’ ’SPF’ .X .Y 28 32 ALL                  SCROLL===> CSR
  ****** **************************** top of data ************************
  =cols> ----+----1----+----2----+----3----+----4----+----5----+----6----+----7--
  000010   RESET #JOBNAME(A8)
  000020   RESET #FD(N3) #FL(A8) #FF(N3)
  000030   RESET #TD(N3) #TL(A8) #TF(N3)
  000040   COMPRESS *INIT-USER ’SM’ INTO #JOBNAME LEAVING NO SPACE
  000050   SET CONTROL ’WL60C6B005/010F’
  000060   INPUT  ’ENTER PARAMETERS FOR LIBRARY COPY:’
  000070    /     ’FROM:   DBID:’ #FD ’FNR:’ #FF ’LIB:’ #FL
  000080    /     ’TO  :   DBID:’ #TD ’FNR:’ #TF ’LIB:’ #TL
  000090 // #JOBNAME JOB JWO,MSGCLASS=X,CLASS=G,TIME=1400
  000100 /*JOBPARM LINES=2000
  000110 //COPY EXEC PGM=NATBAT21,REGION=2000K,TIME=60,
  000120 // PARM=(’DBID=9,FNR=33,FNAT=(,15),FSIZE=19’,
  000130 //       ’EJ=OFF,IM=D,ID=’’;’’,MAINPR=1,INTENS=1’)
  .X     //STEPLIB  DD  DISP=SHR,DSN=OPS.SYSF.V5.ADALOAD
  000150 //         DD  DISP=SHR,DSN=OPS.SYSF.V101.LOAD
  000160 //         DD  DISP=SHR,DSN=OPS.SYSF.PROD.INST  * OPS INSTALL
  .Y     //         DD  DISP=SHR,DSN=OPS.SYSF.SOURCE     * OPS DOCUMENTS
  000180 //DDCARD   DD  *
  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```

When you press **Enter** to invoke the command entered on the COMMAND line of the screen above, the screen appears as shown below:


```
S*>>EDIT-NAT:NSPF101(JOB1JCL)-Program->Report-Free-29K 00004 char  'OPS'
  changed
   COMMAND===>                                                SCROLL===> CSR
  ****** ***************************** top of data ****************************
  =cols> ----+----1----+----2----+----3----+----4----+----5----+----6----+----7--
  000010   RESET #JOBNAME(A8)
  000020   RESET #FD(N3) #FL(A8) #FF(N3)
  000030   RESET #TD(N3) #TL(A8) #TF(N3)
  000040   COMPRESS *INIT-USER 'SM' INTO #JOBNAME LEAVING NO SPACE
  000050   SET CONTROL 'WL60C6B005/010F'
  000060   INPUT  'ENTER PARAMETERS FOR LIBRARY COPY:'
  000070    /     'FROM:   DBID:' #FD 'FNR:' #FF 'LIB:' #FL
  000080    /     'TO  :   DBID:' #TD 'FNR:' #TF 'LIB:' #TL
  000090 // #JOBNAME JOB JWO,MSGCLASS=X,CLASS=G,TIME=1400
  000100 /*JOBPARM LINES=2000
  000110 //COPY EXEC PGM=NATBAT21,REGION=2000K,TIME=60,
  000120 // PARM=('DBID=9,FNR=33,FNAT=(,15),FSIZE=19',
  000130 //       'EJ=OFF,IM=D,ID='';'',MAINPR=1,INTENS=1')
  .X     //STEPLIB  DD  DISP=SHR,DSN=SPF.SYSF.V5.ADALOAD
  ==chg> //         DD  DISP=SHR,DSN=SPF.SYSF.V101.LOAD
  ==chg> //         DD  DISP=SHR,DSN=SPF.SYSF.PROD.INST  * OPS INSTALL
  .Y     //         DD  DISP=SHR,DSN=SPF.SYSF.SOURCE     * OPS DOCUMENTS
  000180 //DDCARD   DD  *
  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```

All occurrences of the string **ops** have been changed to the string **SPF** between lines 140 and 170 and within the Columns 28 to 32.

REPEAT FIND and REPEAT CHANGE Commands

When changing character strings, good use can be made of the **REPEAT** FIND and **REPEAT** CHANGE feature, for instance the sequence:

```
FIND 'abc'
CHANGE 'abc' 'def'
RFIND
RCHANGE
```

allows you to find occurrences of a certain string and optionally change them with relatively small effort.

These commands can also be assigned to PF keys.

Center Data

The Editor also provides commands with which you can center specified data within set boundaries. For example, the sequence:

```
BNDS 5 60;CENTER 5 15
```

centers data in Lines 5 to 15 between Columns 5 and 60.

Note:
Only text already within the boundaries is centered. Text to the left and right of the boundaries is not affected.

Alternatively, you can use certain line commands to perform the centering function:

Line Command	Explanation
TC	Centers this line within the set boundaries.
TCC	Marks the first line in a block of data to be centered. Requires a second TCC line command to delineate the block. The centering is performed after the second TCC command is issued.

Copy, Move, Overlay and Repeat Lines

With the following line commands, you can copy, move or repeat lines or blocks of data. **M**ove and **C**opy line commands can only be used in conjunction with an After (**A**), Before (**B**) or an Overlay (**O**) command.

Line Command	Explanation
A	Marks target line for **C**opy and **M**ove line commands. Data are inserted after this line.
B	Marks target line for **C**opy and **M**ove line commands. Data are inserted before this line.
C	Copy this line to position marked by an **A**, **B** or **O** line command.
C15	Copy the next 15 lines to position marked by an **A**, **B** or **O** line command.
CC	Marks the first line of the block to be copied. A second **CC** is required to the delineate block. Copying is performed after target has been marked with an **A**, **B** or **O** line command.
CX	Copies the line labelled **.X**. Inserts data after this line.
CY	Copies the line labelled **.Y**. Inserts data after this line.
CX-Y	Copies the block of lines from the line labelled **.X** to the line labelled **.Y**. Inserts data after this line.
M	Move this line to position marked by an **A**, **B** or **O** line command.
M15	Move the next 15 lines to position marked by an **A** or **B** line command.
MM	Marks the first line of the block to be moved. A second **MM** is required to delineate the block. The move is performed after the target has been marked with an **A**, **B** or **O** line command.
MX	Moves the line labelled **.X**. Inserts data after this line.
MY	Moves the line labelled **.Y**. Inserts data after this line.
MX-Y	Moves the block of lines from the line labelled **.X** to the line labelled **.Y**. Inserts data after this line.
O	Marks target line for **C**opy or **M**ove line commands. Data are merged with this line. (Only blank characters **within boundaries** are changed).
O15	Merges first 15 lines of a block defined by a **C**opy or **M**ove line command with the next 15 lines. (Only blank characters **within boundaries** of the target lines are changed).
OO	Marks the first line of a block to be merged with a block of data defined by a **C**opy or **M**ove line command. A second **OO** command is required to delineate target block. (Only blank characters **within boundaries** are changed).
R	Repeat this line once.
R15	Repeat this line 15 times.
RR	Marks the first line of a block to be repeated. A second **RR** is required to delineate the block. The repeat is performed after second **RR** is entered.

RR15	Repeat block delineated by two **RR15** line commands 15 times. The **parenthesis line commands**, below, move text the full number of columns specified, but only **within the set boundaries**, therefore, part of the moved text could disappear.
)	Move this line right by one column beginning with left boundary.
)15	Move this line right by 15 columns.
))15	Marks first line of a block to be moved right by 15 columns. A second **))15** is required to delineate the block. The move is performed after the second **))15** command is entered. Two unqualified **))** line commands move the block right by 1 column.
(	Move this line left by one column.
(15	Move this line left by 15 columns.
((15	Marks first line of a block to be moved left by 15 columns. A second **((15** is required to delineate the block. The move is performed after the second **((15** command is entered. Two unqualified **((** line commands move the block left by 1 column. If you use the **greater than** or **less than** symbols, below, to move data, the maximum move possible is up to the next non-blank character **within the set boundaries**.
>	Move this line right by one column.
>15	Move this line right by 15 columns.
>>15	Marks first line of a block to be moved right by 15 columns. A second >> is required to delineate the block. The move is performed after the second >> command is entered. Two unqualified >> line commands move the block right by 1 column.
<	Move this line left by one column.
<15	Move this line left by 15 columns.
<<15	Marks first line of a block to be moved left by 15 columns. A second << is required to delineate the block. The move is performed after the second << command is entered. Two unqualified << line commands move the block left by 1 column.

Example of Overlaying Data

The **O** line command allows you to merge single-column lists into multi-column format (i.e. tabular form). You can use the **O** line command in conjunction with the **M** (Move) or **C** (Copy) line commands.

The following two figures illustrate this function:

```
S*>>EDIT-NAT:NSPF101(JOB1JCL)-Program->Report-Free-30K--------- columns 001 072
  COMMAND===>                                                   SCROLL===> CSR
  ****** ***************************** top of data ****************************
  000090 //JOBNAME JOB JWO,MSGCLASS=X,CLASS=G,TIME=1400
  000100 /*JOBPARM LINES=2000
  000110 //COPY EXEC PGM=NATBAT21,REGION=2000K,TIME=60,
  000120 // PARM=('DBID=9,FNR=33,FNAT=(,15),FSIZE=19',
  000130 // 'EJ=OFF,IM=D,ID='';'',MAINPR=1,INTENS=1')
  ooO140 //STEPLIB DD
  000150 //        DD
  000160 //        DD
  oo0170 //        DD
  MM0180                    DISP=SHR,DSN=SPF.SYSF.V5.ADALOAD
  000190                    DISP=SHR,DSN=SPF.SYSF.V101.LOAD
  000200                    DISP=SHR,DSN=SPF.SYSF.PROD.INST   * OPS INSTALL
  MM0210                    DISP=SHR,DSN=SPF.SYSF.SOURCE      * OPS DOCUMENTS
  000220 //DDCARD  DD *
  000230 ADARUN DA=9,DE=3380,SVC=249
  000240 //CMPRINT DD SYSOUT=X
  000250 //CMPRT01 DD SYSOUT=X
  000260 //CMWKF01 DD DUMMY
  000270 //CMSYNIN DD *
  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up     Down  Swap  Left  Right Curso
```

In the figure above, lines 180 to 210 are marked with the **MM** (Move) line command. They are to be overlaid on lines 140 to 170, which are marked with the **OO** (Overlay) line command.

This figure shows the result of the line commands displayed in the first figure. Lines 180 to 210 have been overlaid on Lines 140 to 170:

```
S*>>EDIT-NAT:NSPF101(JOB1JCL)-Program->Report-Free-30K--------- columns 001 072
   COMMAND===>                                                  SCROLL===> CSR
  ****** ***************************** top of data ****************************
  000090 //JOBNAME JOB JWO,MSGCLASS=X,CLASS=G,TIME=1400
  000100 /*JOBPARM LINES=2000
  000110 //COPY EXEC PGM=NATBAT21,REGION=2000K,TIME=60,
  000120 // PARM=('DBID=9,FNR=33,FNAT=(,15),FSIZE=19',
  000130 //       'EJ=OFF,IM=D,ID='';'',MAINPR=1,INTENS=1')
  000140 //STEPLIB  DD           DISP=SHR,DSN=SPF.SYSF.V5.ADALOAD
  000150 //         DD           DISP=SHR,DSN=SPF.SYSF.V101.LOAD
  000160 //         DD           DISP=SHR,DSN=SPF.SYSF.PROD.INST  * OPS INSTALL
  000170 //         DD           DISP=SHR,DSN=SPF.SYSF.SOURCE     * OPS DOCUMENTS
  000220 //DDCARD   DD  *
  000230 ADARUN DA=9,DE=3380,SVC=249
  000240 //CMPRINT  DD  SYSOUT=X
  000250 //CMPRT01  DD  SYSOUT=X
  000260 //CMWKF01  DD  DUMMY
  000270 //CMSYNIN  DD  *
  000280 LOGON SYSMAIN2
  000290 CMD C C * FM  #FL DBID  #FD FNR  #FF TO  #TL DBID  #TD FNR  #TF REP
  000300 FIN
  000310 /*
  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```


Copy a Window with Data

You can specify a window with data for move or copy operations. This allows you to copy or move data that does not start or end at the beginning or end of a line. This function can be performed using Editor line commands and/or main commands.

When you define a window, all data on your screen between start and end of the window become part of the window.

Line Commands Available

Line Command	Explanation
WS	Marks start of data window. Cursor position marks the column from which data are read. If the cursor is not in the line for which the command is entered, the window starts in column 1.
WS*n*	Data window starts in Column **n** of this line.
WE	Marks end of data window. Works in the same way as **WS**. If the window is to start and end in the same line, overtype the **WS** command with the **WE** command. The Editor acknowledges the set window with messages **WS** (**WSn**) and **WE** (**WEn**) in the line command field, or with **WW** if the start and end of the window are in the same line. Before defining a new window, reset the old window with the **RESET** command to avoid a command conflict.
WE*n*	Data window ends in Column **n** of this line.
WC	Copies the data window. The cursor position marks the column at which this line is to be split to insert the copied data.
WC*n*	Splits this line in Column **n**, and copies the data between the two parts of the line.
WM	Moves the data window. Works in the same way as **WC**, but the original data is deleted after the copy operation.
WM*n*	Splits this line in Column **n**, and moves the data between the two parts of the line.

Main Commands Available

Main Command	Explanation

WINDOW	Defines a window. The starting line and column, and the end line column are specified in the command parameters. At least one parameter is required. Examples:
	`WINDOW 5 10 24 13`
	Defines a window starting in Line 5 / Column 24, and ending in Line 10, Column 13.
	`WINDOW 5 10 24`
	Defines a window starting in Line 5 / Column 24, and ending in Line 10 / last column.
	`WINDOW 5 10`
	Defines a window starting in Line 5 / first column, and ending in Line 10 / last column.
	`WINDOW 5 5`
	Defines a window starting in line 5 / first column, and ending in line 5 / last column.

CWINDOW	Copy a window defined with the WINDOW command. Optional parameters specify the line at which the window is to be inserted. Examples:
	`CWINDOW 5`
	Copies the window after Line 5.
	`CWINDOW 5 24`
	Splits Line 5 at Column 24 and copies the window in between the two parts.

DWINDOW	Deletes a window of data defined by the WINDOW command.
MWINDOW	Moves window defined by the WINDOW command. Works like the CWINDOW command, but data in the original window is deleted after the copy operation.

Example:

Consider the following text:

```
EDIT-PDS:MBE.COMN.SOURCE(WINEX) ----------------------------  Columns 001 072
   COMMAND===>                                                  SCROLL===> CSR
  ****** ***************************** top of data ****************************
  000001 Copy a Window with Data
  000002
  000003 You can specify a window with data for move or copy operations. This
  000004 allows you to copy or move data that does not start or end at the
  000005 beginning or end of a line. This function can be performed using
  000006 Editor line commands and/or main commands.
  000007
  000008 Below are some examples of copying windows with data. Note that when
  000009 you define a window, all data on your screen between start and end of
  000010 of the window become part of the window. Available line commands are:
  ****** **************************** bottom of data ****************************









  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```

If you wish to move the whole sentence starting **Note that when...** (Line 8) to follow the first sentence of the displayed text ending **...copy operations** (Line 3).

1. Type the line command **WS** in Line 8, the first line of data to be moved, place the cursor in the required column (**N** of the word **Note**) and press **Enter**. The message **WS55** appears in the prefix area of Line 8, indicating the column number selected:

```
EDIT-PDS:MBE.COMN.SOURCE(WINEX) ----------------------------- Block is pending
   COMMAND===>                                                 SCROLL===> CSR
  ****** ***************************** top of data ****************************
  000001 Copy a Window with Data
  000002
  000003 You can specify a window with data for move or copy operations. This
  000004 allows you to copy or move data that does not start or end at the
  000005 beginning or end of a line. This function can be performed using
  000006 Editor line commands and/or main commands.
  000007
  WS55   Below are some examples of copying windows with data. Note that when
  000009 you define a window, all data on your screen between start and end of
  000010 of the window become part of the window. Available line commands are:
  ****** **************************** bottom of data ****************************









  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```

2. Now type the line command **WE** in Line 10, the last line of data to be moved, move the cursor to the last column to be moved (full stop '**.**' after **window**) and press **Enter**. The message **WE40** appears in the prefix area of Line 10:

```
EDIT-PDS:MBE.COMN.SOURCE(WINEX) ----------------------------- Block is pending
   COMMAND===>                                                 SCROLL===> CSR
  ****** ***************************** top of data ****************************
  000001 Copy a Window with Data
  000002
  000003 You can specify a window with data for move or copy operations. This
  000004 allows you to copy or move data that does not start or end at the
  000005 beginning or end of a line. This function can be performed using
  000006 Editor line commands and/or main commands.
  000007
  WS55   Below are some examples of copying windows with data. Note that when
  000009 you define a window, all data on your screen between start and end of
  WE40   of the window become part of the window. Available line commands are:
  ****** **************************** bottom of data ****************************









  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```

3. Now type the line command **WM** in the line command field of Line 3 (the data will be moved to the following line, Line 4), and move the cursor to the column at which Line 3 is to be split (the blank before the word **This**). Press **Enter**. The following screen illustrates the result:

```
EDIT-PDS:MBE.COMN.SOURCE(WINEX) ------------------------------ Columns 001 072
      COMMAND===>                                               SCROLL===>    CSR
     ****** ***************************** top of data ****************************
     000001 Copy a Window with Data
     000002
     000003 You can specify a window with data for move or copy operations.
     000004 Note that when
     000005 you define a window, all data on your screen between start and end of
     000006 of the window become part of the window.
     000007 This
     000008 allows you to copy or move data that does not start or end at the
     000009 beginning or end of a line. This function can be performed using
     000010 Editor line commands and/or main commands.
     000011
     000012 Below are some examples of copying windows with data.
     000013
                                          Available line commands are:
     ****** *************************** bottom of data ***************************




     Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
           Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```

To achieve the same result using main commands, you must use the following command sequence:

```
WINDOW 8 10 55 40;MWINDOW 3 64
```

Define Horizontal and Vertical Boundaries

The Editor provides commands which allow you to set horizontal and vertical boundaries within which certain functions can be performed.

With the BNDS main command, you can define horizontal boundaries. For example, the command:

```
BNDS 20 50
```

sets horizontal limits at the Columns 20 and 50. Commands which refer to these boundaries include theFIND, CHANGE, CENTER, ORDER, JLEFT, and JRIGHT main commands, as well as their corresponding line commands (TC, TO, LJ, RJ, etc.).

To display the current boundary settings, issue the BNDS line command.
The following figure shows the result of the **BNDS 20 50** main command followed by a BNDS line command in Line 1:


```
S*>>EDIT-NAT:NATLIB1(JOB1JCL)-Program->Report-Free-30K--------- columns 001 072
   COMMAND===>                                                  SCROLL===> CSR
  ****** ***************************** top of data ******************************
  =bnds>                   <                            >
  000010   RESET #JOBNAME(A8)
  000020   RESET #FD(N3) #FL(A8) #FF(N3)
  000030   RESET #TD(N3) #TL(A8) #TF(N3)
  000040   COMPRESS *INIT-USER 'SM' INTO #JOBNAME LEAVING NO SPACE
  000050   SET CONTROL 'WL60C6B005/010F'
  000060   INPUT  'ENTER PARAMETERS FOR LIBRARY COPY:'
  000070    /     'FROM:   DBID:' #FD 'FNR:' #FF 'LIB:' #FL
  000080    /     'TO  :   DBID:' #TD 'FNR:' #TF 'LIB:' #TL
  000090 // #JOBNAME JOB JWO,MSGCLASS=X,CLASS=G,TIME=1400
  000130 /*JOBPARM LINES=2000
  000140 //COPY EXEC PGM=NATBAT21,REGION=2000K,TIME=60,
  000150 // PARM=('DBID=9,FNR=33,FNAT=(,15),FSIZE=19',
  000160 //       'EJ=OFF,IM=D,ID='';'',MAINPR=1,INTENS=1')
  000170 //STEPLIB  DD  DISP=SHR,DSN=OPS.SYSF.V5.ADALOAD
  000180 //         DD  DISP=SHR,DSN=OPS.SYSF.PROD.LOAD
  000190 //DDCARD   DD  *
  000200 ADARUN DA=9,DE=3380,SVC=249
  000210 //CMPRINT  DD  SYSOUT=X
  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```

With the LABEL main command, you can label the current line (line currently at the top of edit area), or a block of lines. For example, the command:

```
LABEL  .X
```

labels the current line **.X**. To delineate a block of lines, you can now scroll to the next required delimiting line and issue a LABEL ***.Y*** command. The new current line is labelled **.Y**. You can also perform the labelling function with line commands. Simply enter **.X** and **.Y** directly in the line command fields of the required lines. You can use any string to label lines, for example **.START** and **.END**.

For examples of the use of labelled lines, see the sections Find a Specific Character String and Change a Specific Character String.

Define a Mask Line

You can define data that is automatically placed in a line added through a line insertion operation (for example, using the line command **I** or **W**). Such a line is referred to as a **mask line** and is defined by typing the line command MASK in the line command field, pressing **Enter**, and entering the required data in the new line. This mask line is indicated by the message ***=mask*** in the line number field when you press **Enter**.

Note: You can only have one mask line during an edit session. If you define a new mask line, any existing mask line definition is automatically updated with the new value.

When you have defined a mask line, you must activate the mask feature using the command:

```
MASK ON
```

The defined mask line now appears in all lines added through a line insert operation (but remember that any line added by a line insert operation is deleted if no data or blank is added). This feature is useful if you often type the same or similar line.

To deactivate the mask feature, issue the main command:

```
MASK OFF
```

The defined mask line remains in force for the duration of the edit session. This means that you can reactivate the mask line with the MASK main command.

Display Boundary, Tab and Column Positions

You can display the positions of your boundaries (set with a BNDS main command) and tabs (set by a TABS main command), as well as the edit area column positions on any line by using the appropriate line command as listed below:

Line Command	Explanation
BNDS	Displays the boundary positions on this line.
COLS	Displays the column positions on this line.
TABS	Displays the tab positions on this line.

Exclude/Include Lines from Display

You can exclude specific lines from the display using the EXCLUDE main command. For example, the command:

```
EXC 'ABC' .X .Y ALL
```

excludes all lines with the string **ABC** within the block labelled **.X** and **.Y** from display. An unqualified EXCLUDE command excludes the current line. Each excluded line or block of lines is replaced by a line of dashes and a message informing you how many lines are excluded.

To recall excluded lines to display, use the INCLUDE main command. For example, the command:

```
IN C'Abc' ALL
```

includes all excluded lines containing the string **Abc** exactly as entered here. An unqualified INCLUDE command recalls the first line in the excluded block.

The EXCLUDE and INCLUDE main commands can be issued with the same string and search operands as described for the FIND command, except that the ALL search direction operand means **exclude/include all lines with the given string**.

Lines can also be excluded or recalled to display using any of the line commands listed below.

Line Command	Explanation
X	Exclude this line from display.
X15	Exclude the next 15 lines from display.
XX	Marks the first line of a block of data to be excluded from display. A second **XX** line command is required to delineate the block. The exclusion is performed after the second **XX** is entered.
F	Recall this line to display or recall the first line of the excluded block to display.
F15	Recall the first 15 lines of the excluded block to display.
L15	Recall the last 15 lines of the excluded block to display.

You can issue the main command XSWAP to exchange excluded lines with displayed lines.

Find a Specific Character String

To locate a specific character string, you can use the FIND main command with operands defining the string, the area to be searched and the direction of search.The cursor is placed on the first character of the string. If the line containing the string was excluded from display, it is now included in the display.

The following sections describe the possible command operands.

String Definition Operand

The string operand defines the character string to be located.
You can specify any of the following:

Operand	Explanation
*	Find string specified in previous FIND command.
’abc’	Find string **abc** regardless of whether the string is upper case or lower case.
C’Abc’	Find string exactly as entered here.
P’a(char)c’	Find string whose first character is ***a*** and third character is **c**. **(char)** stands for a special character acting as a wildcard character with the following meaning: **=** - any character **§** - alphabetic character **#** - numeric character **$** - special character ***** - non-blank character **-** - non-numeric character **<** - lower-case character **>** - upper-case character
T’abc’	Find string **abc** regardless of whether the string is upper case or lower case.
X’D4A8’	Find string that corresponds to hexadecimal D4A8.

String-Matching Operand

The string-matching operand specifies whether any special occurrence of the string is to be located. The following options are possible:

Operand	Explanation
CHARS	No restrictions (any occurrence of the string).
PREFIX	Only those occurrences which are the prefix of a word.
SUFFIX	Only those occurrences which are the suffix of a word.
WORD	Only those occurrences which form a word.

Default is CHARS.

Direction Operand

The direction operand specifies the direction of the search operation.
The following options are possible:

Operand	Explanation
ALL	Any occurrence of the string (search all directions).
FIRST	First occurrence of the string.
LAST	Last occurrence of the string.
NEXT	Next occurrence of the string starting from the cursor position.
PREV	Previous occurrence of the string.

Default is NEXT.

Line-Type Operand

This line-type operand specifies whether excluded or included lines only are to be searched. The following options are possible:

Operand	Explanation
X	Search excluded lines only.
NX	Search non-excluded lines only.

If this operand is omitted, the Editor searches all data for the given string, included and excluded lines. If the string is found in an excluded line, it is returned to display.

Block Operand

If you have labelled lines or a block of lines, you can use the block operand to restrict the search area for the FIND command.
Two examples of the block operand follow:

Operand	Explanation
.X	Search from line labelled **.X** to end of data.
.X .Y	Search from line labelled **.X** to line labelled **.Y**.

where **X** and **Y** can be any alphabetic character or four-character string.

Columns Operand

The column operand allows you to restrict the search for the given string between certain columns. Below are two examples of the columns operand.

Operand	Explanation
20	Locate given string starting in column 20 (the first character of the string must be in column 20).
20 40	Locate given string anywhere between columns 20 to 40.

Examples of FIND Command

Command	Explanation
F C'HILITE' X PREV	Find the previous occurrence of the string **HILITE** exactly as entered here; search excluded lines only.
F P'RCV#' .X .Z 20 30	Find the string starting **RCV** with a numeric fourth character within the block **.X .Z** and between columns 20 to 30.
F X'6C' SUFFIX NX	Find the character corresponding to the hexadecimal **6C** in non-excluded lines only. The character must end a word.

If single quotation marks are part of the string to be found, you must use a different separator in the FINDcommand, for example double quotation marks:
FIND C"'string'".

You can repeat a previous FIND command with the **R**FIND main command.

Insert/Delete Lines

The following line commands are available to insert and delete lines on your Editor screen.

Line Command	Explanation
D	Deletes this line.
D15	Deletes the next 15 lines.
DD	Marks the first line of a block to be deleted. A second **DD** line command is required to delineate the block. The deletion is performed after the second **DD** is entered.
DX	Deletes the line labelled **.X**.
DY	Deletes the line labelled **.Y**.
DX-Y	Deletes the block of lines from the line labelled **.X** to the line labelled **.Y**.
I	Inserts one line after this one. The Editor switches to insert mode. This means if you type data or enter a blank on the new line and press **Enter**, a new line is automatically inserted and the cursor placed in it. If you enter no new data in an inserted line and press **Enter**, the Editor leaves insert mode and the blank line is deleted. If you have defined a mask line and the **MASK** setting is **ON**, the mask line is inserted when you issue this command.
I15	Inserts 15 lines after this one. You can type data in the new lines. When you press **Enter**, unused lines are deleted but one blank line remains with the cursor in it (Editor stays in insert mode).
TE	Switches the Editor to **text enter mode**. This means that beginning with this line the Editor screen is blank (without line numbers) and you can enter data. When you press **Enter**, any remaining blank lines are deleted, the line numbers are re-displayed and the text is reformatted within the set margins and with the specified justification. See also the following section **POWER Command**.
W	Opens a window of one line. No new line is inserted if you enter data in the window line and press **Enter**.
W15	Opens a window of 15 lines. When you press **Enter**, all unused lines are deleted.

POWER Command

You can also use the POWER main command to switch to **text enter mode**.

When you issue the POWER main command, you are presented with a blank Editor screen (without line numbers) and you can enter data. When you press **ENTER**, any remaining blank lines are deleted, the line numbers are redisplayed and the text is reformatted within the set margins and with the specified justification.

DELETE Command

Line deletion can also be performed with the DELETE main command.

For example, the command:

```
DEL C'Abc' .X .Y 10 30 ALL
```

deletes all lines containing the string **Abc** exactly as entered here between columns 10 to 30 within the block delineated by the labels **.X** and **.Y**.

You can specify all operands described for the FIND command above, except that the ALL direction operand specifies deletion of all lines with the given string. An unqualified DELETE command deletes the current line.

Justify Data

A number of main commands and line commands are available to rearrange lines or blocks of data on your screen, depending on the setting of your horizontal boundaries (BNDS main command); see the section Define Horizontal and Vertical Boundaries.

The JLEFT and JRIGHT main commands justify the specified data with the left and right boundaries respectively. For example, the sequence:

```
BNDS 16 80;JLEFT 14 17
```

justifies the data between columns 16 to 80 in lines 14 to 17 with column 16.

The figure below illustrates this example:


```
S*>>EDIT-NAT:NSPF101(JOB1JCL)-Program->Report-Free-29K--------- columns 001 072
   COMMAND===> BNDS 16 80;JLEFT 14 17                            SCROLL===> CSR
  ****** ***************************** top of data ****************************
  =cols> ----+----1----+----2----+----3----+----4----+----5----+----6----+----7--
  000010   RESET #JOBNAME(A8)
  000020   RESET #FD(N3) #FL(A8) #FF(N3)
  000030   RESET #TD(N3) #TL(A8) #TF(N3)
  000040   COMPRESS *INIT-USER 'SM' INTO #JOBNAME LEAVING NO SPACE
  000050   SET CONTROL 'WL60C6B005/010F'
  000060   INPUT  'ENTER PARAMETERS FOR LIBRARY COPY:'
  000070    /     'FROM:   DBID:' #FD 'FNR:' #FF 'LIB:' #FL
  000080    /     'TO  :   DBID:' #TD 'FNR:' #TF 'LIB:' #TL
  000090 // #JOBNAME JOB JWO,MSGCLASS=X,CLASS=G,TIME=1400
  000100 /*JOBPARM LINES=2000
  000110 //COPY EXEC PGM=NATBAT21,REGION=2000K,TIME=60,
  000120 // PARM=('DBID=9,FNR=33,FNAT=(,15),FSIZE=19',
  000130 //       'EJ=OFF,IM=D,ID='';'',MAINPR=1,INTENS=1')
  000140 //STEPLIB  DD           DISP=SHR,DSN=SPF.SYSF.V5.ADALOAD
  000150 //         DD           DISP=SHR,DSN=SPF.SYSF.V101.LOAD
  000160 //         DD           DISP=SHR,DSN=SPF.SYSF.PROD.INST  * OPS INSTALL
  000170 //         DD           DISP=SHR,DSN=SPF.SYSF.SOURCE     * OPS DOCUMENTS
  000180 //DDCARD   DD  *
  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```

The following screen shows the result of the command displayed in the command line of the first screen:

```
S*>>EDIT-NAT:NSPF101(JOB1JCL)-Program->Report-Free-29K--- file has been ordered
   COMMAND===> BNDS 16 80;JLEFT 14 17                            SCROLL===> CSR
  ****** ***************************** top of data ****************************
  =cols> ----+----1----+----2----+----3----+----4----+----5----+----6----+---
  000010   RESET #JOBNAME(A8)
  000020   RESET #FD(N3) #FL(A8) #FF(N3)
  000030   RESET #TD(N3) #TL(A8) #TF(N3)
  000040   COMPRESS *INIT-USER 'SM' INTO #JOBNAME LEAVING NO SPACE
  000050   SET CONTROL 'WL60C6B005/010F'
  000060   INPUT  'ENTER PARAMETERS FOR LIBRARY COPY:'
  000070    /     'FROM:   DBID:' #FD 'FNR:' #FF 'LIB:' #FL
  000080    /     'TO  :   DBID:' #TD 'FNR:' #TF 'LIB:' #TL
  000090 // #JOBNAME JOB JWO,MSGCLASS=X,CLASS=G,TIME=1400
  000100 /*JOBPARM LINES=2000
  000110 //COPY EXEC PGM=NATBAT21,REGION=2000K,TIME=60,
  000120 // PARM=('DBID=9,FNR=33,FNAT=(,15),FSIZE=19',
  000130 //       'EJ=OFF,IM=D,ID='';'',MAINPR=1,INTENS=1')
  000140 //STEPLIB  DD  DISP=SHR,DSN=SPF.SYSF.V5.ADALOAD
  000150 //         DD  DISP=SHR,DSN=SPF.SYSF.V101.LOAD
  000160 //         DD  DISP=SHR,DSN=SPF.SYSF.PROD.INST  * OPS INSTALL
  000170 //         DD  DISP=SHR,DSN=SPF.SYSF.SOURCE     * OPS DOCUMENTS
  000180 //DDCARD   DD  *
  Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
        Help  Split End   Suspe Rfind Rchan Up    Down  Swap  Left  Right Curso
```

The sequence:

```
BNDS 10;JRIGHT 15
```

justifies the data on the right of Column 10 in lines 15 to the end of the screen with the last column of your Editor screen (Column 88 of your terminal screen).

Alternatively, you can justify lines or blocks of data using any of the line commands listed in the following table:

Line Command	Explanation
LJ	Justifies the data within the set boundaries in this line with the left boundary.
LJJ	Marks the first line of a block of data within the boundaries (set with the BNDS main command) to be justified to the left. A second **LJJ** line command is required to delineate the block. Justification is performed after the second **LJJ** is issued.
RJ	Justifies the data within the set boundaries in this line with the right boundary.
RJJ	Marks the first line of a block of data within the boundaries (set with the BNDS main command) to be justified to the right. A second **RJJ** line command is required to delineate the block. Justification is performed after the second **RJJ** is issued.

You can also justify data to the left boundary, or order data between left and right boundary in conjunction with the JUSTIFY command.

For example, the command sequence:

```
BNDS 10 60;JUSTIFY LEFT
```

enables justification to the left boundary. Mark a block of data with two TOO line commands (explained below) to reformat the data between Columns 10 and 60, justified to Column 10.

Locate a Specific Line

If you wish to display a specific line at the top of your Editor screen (i.e. make it the current line), use the LOCATE main command with a parameter describing the line you wish to become the current line.

Examples:

- The main command **L 32** makes Line 32 the current line;
- The main command **32** makes Line 32 the current line;
- The main command **L .X** makes the line labelled **.X** the current line;
- The main command **L 'ABC'** makes the first line that starts with the string **ABC** the current line (useful when browsing sorted data such as directory lists).

Differences between the LOCATE and FIND Commands

Please note the following differences between the LOCATE and FIND commands:

- If you issue the LOCATE command with a character string (**L 'ABC'**), the string is only found if it starts in column 1. The FIND command searches the whole data area;
- With the LOCATE command, it is assumed that the data to be searched is sorted in ascending alphabetical order;
- When a line is located with the LOCATE command, the cursor is placed in the prefix area; with the FIND command, the cursor is placed on the found string and the line is not necessarily made the current line.

Order Data Between Specified Boundaries

You can change the indentation of specified lines using the ORDER main command together with a boundary setting. For example, the command sequence:


```
BNDS 3;ORDER 5 20
```

moves lines 5 to 20 right to start in Column 3.

Note:
If the end of any ordered line traverses the right boundary, it is automatically split.

To re-justify the shifted data to the left, use a JLEFT command.

You can also change the indentation of lines or of a block of data using certain line commands. Here, too, if the end of any line traverses the right boundary, it is automatically split.

Line Command	Explanation
TF	Orders data from the line on which it was entered to the end of the paragraph or to the next blank line with the right boundary. This line command can be entered with a numerical value specifying the right boundary, e.g. the line command ***TF50*** orders data with Column 50.
TO	Marks one line to be ordered.
TOO	Marks the first line of a block of data to be ordered. Requires a second TOO line command to delineate the block. Ordering is performed after the second TOO is issued.

Data can be ordered within set boundaries and justified to the left boundary, right boundary or both using the JUSTIFY command. For example, the command:

```
BNDS 6 60;JUSTIFY BOTH
```

activates justification to Columns 5 and 60. To perform the ordering, mark a block of data with two TOO line commands.

The Editor also provides a line command with which you can split a single line into two. Type the line command ***S*** in the command field of the line you wish to split, move the cursor to the position where the split is to occur and press **Enter**.

Scroll Data on the Editor Screen

PF Keys

Commands for scrolling data are often assigned to the following PF keys:

- **PF7** (main command UP) to scroll toward top of data.
- **PF8** (main command DOWN) to scroll toward bottom of data.
- **PF10** (main command LEFT) to scroll data to the left
- **PF11** (main command RIGHT) to scroll to the right.

Settings for SCROLL Field

In the SCROLL field at the top right of your screen, you can enter scroll settings. These settings are used to set the scroll amount for the PF keys, above, and some are also used with the scrolling main commands on the following page.

Possible settings for the SCROLL field are:

Scroll Setting	Explanation
<number>	Scroll up or down a specified number of lines. Scroll right or left a specified number of columns.
CSR (default)	Scroll down to cursor position if cursor is on a line of text. Cursor line becomes first line of text. When scrolling up, cursor line becomes last line of text. Scroll a page length, if cursor is in COMMAND line. Scroll right or left to cursor position.
DATA	Scroll a page length minus one line. When scrolling down, the bottom line becomes the top line. When scrolling up, the top line becomes the bottom line. When scrolling right, the last column becomes the first column. When scrolling left, the first column becomes the last column.
HALF	Scroll half a page in any direction.
LINE	Scroll **up** to beginning of line or **down** to end of line.
MAX	Scroll to top or bottom of data. Scroll to extreme right or left of data.
PAGE	Scroll a page length in any direction.
PARA	Scroll up or down to first character of next paragraph.
SENT	Scroll **up** to first character of current sentence or **down** to first character of following sentence. When scrolling up, if cursor is on first character of sentence, scroll to first character of previous sentence.
WORD	Scroll **up** to first character of next word or **down** to first character of following word.

Main Commands for Scrolling

Apart from the LOCATE main command which scrolls data to a specified line, several main commands are available for vertical and horizontal scrolling.

The following table shows all possible scrolling commands and their meaning:

Main Command	Explanation
BOTTOM *or* ++	Scrolls to the end of the object being edited.
TOP *or* - -	Scrolls to the beginning of the object being edited.
DOWN	Scrolls forward by the amount specified in the SCROLL field.
DOWN *n*	Scrolls forward by *n* lines.
+*n*	Scrolls forward by *n* lines.
UP	Scrolls backwards by the amount specified in the SCROLL field.
UP *n*	Scrolls backwards by *n* lines.
-*n*	Scrolls backwards by *n* lines.
LEFT	Scrolls to the left by the amount specified in the SCROLL field.
LEFT *n*	Scrolls to the left by *n* columns.
RIGHT	Scrolls to the right by the amount specified in the SCROLL field.
RIGHT *n*	Scrolls to the right by *n* columns.
FIX *n*	Specifies the number of columns *n*, starting with Column 1, to remain in display when scrolling to the right.

Sort Lines in Alphabetical Order

You can sort data lines in ascending or descending alphabetical order according to sorting criteria. For example, the command:

```
SORT 10 15
```

sorts all lines in the member in ascending order according to the characters beginning in Column 10 and ending in Column 15.

To sort only a block of lines, for example, label the lines where the block is to start and end with **.X** and **.Y** respectively. The command:

```
SORT .X .Y D
```

sorts all lines in the block marked by **.X** and **.Y** in descending order.

To sort a block of lines according to the characters beginning in Column 5 and ending in Column 20, for example, label the lines where the block is to start and end with **.X** and **.Y** respectively. The command:

```
SORT 5 20 .X .Y
```

sorts all lines in the block marked by **.X** and **.Y** in ascending order according to the characters beginning in Column 5 and ending in Column 20.

Store Data / Leave the Editor

You can store data and/or leave the Editor using any of the following main commands:

Main Command	Explanation
CANCEL	Leaves the Editor. Any changes made during this edit session do not take effect.
END	If AUTOSAVE is ON, stores data including any changes, and leaves the Editor. If AUTOSAVE is OFF, the END main command acts as a CANCEL command if no data was modified. If changes were made, a message asks you to issue a SAVE or CANCEL command.
SAVE	Stores the data, including any changes. Editing session continues.

Use the Physical/Logical Tabulator

TABS main command

When you issue the TABS ON main command, the **standard tab positions** set in your Editor profile are turned **on** and **tabs on std** appears in your profile. Issue the TABS OFF main command to turn tabulation **off** again.

Setting Standard Tab Positions

To turn tabulation **on** and set the **standard tab positions** for your profile to Columns 10, 20, 30, 40 and 50, for example, issue the main command **TABS 10 20 30 40 50**.

Setting the Logical Tab Character

To turn tabulation **on** and set the **logical tab character** to **%**, for example, issue the main command **TABS %**.

You can enter data and automatically move it to a specific tab position by preceding it with a logical tab character. One tab character moves the data to the next tab position, two tab characters moves the data to the second tab position etc.

Setting Justification Parameters

Apart from tab positions, you can specify the following parameters with the TABS main command:

Parameter	Explanation
DECIMAL	Orders data with the decimal point in the data at the tab position.
LEFT	Orders data to the left of the tab position.
RIGHT	Orders data to the right of the tab position.

For further information, see the Examples 1, 2 and 3 below.

To display the current logical tab character and shift parameter (excluding tab positions), issue the PROFILE main command.

TABS Line Command

When you issue the TABS line command in any line, the current tab positions set in your Editor Profile are displayed in that line and marked with asterisks ***** if no logical tab character has been set. This command does not turn tabulation on.

For example, issue the TABS line command to display the positions set with the main command TABS 10 20 30 40 50.

This displays the current tab positions as follows:

```
=tabs>              *         *         *         *         *         *
```

Setting Multiple Logical Tab Characters and Mixed Justification Parameters

To tabulate data in a specific column and with a specific shift, multiple logical tab characters and mixed justification parameters are possible.

To set the multiple logical tab characters, issue the TABS line command and overtype each asterisk ***** with a special character. Any data typed in preceded by any of these logical tab characters are tabulated in the corresponding column.

To set the mixed justification parameters, type **L** (Left), **R** (Right) or **D** (Decimal) to the right of each logical tab character for **left**, **right** or **decimal** ordering.

For further information, see the Examples 4 and 5 below.

Examples:

Below are some examples of tabulation (the percentage sign **%** is assumed to be the tabulation character; the COLS line command has been issued to display column position):

Example 1: Tab Positions

The command:

```
TABS 10 20 40 LEFT
```

activates logical tabs with Tabulation Columns 20, 30 and 40 with left justification. After you press **Enter**, the input text line:

```
000010 %abc      %def            %ghi
```

is displayed as follows:

```
=cols> ----+----1----+----2----+----3----+----4----+----5----+----6
               abc       def               ghi
```

Example 2: TABS RIGHT

The command:

```
TABS RIGHT
```

activates logical tabs with right justification. After you press **Enter**, the input text line:

```
000010 %abc      %def            %ghi
```

is displayed as follows:

```
=cols> ----+----1----+----2----+----3----+----4----+----5----+----6
             abc       def               ghi
```

Example 3: TABS DECIMAL

The command:

```
TABS DECIMAL
```

activates logical tabs with justification of the decimal point in the tab position. After you press **Enter**, the input text line:

```
000010 %15.27$ %16.3 DM    %13 IS
```

is displayed as follows:

```
=cols> ----+----1----+----2----+----3----+----4----+----5----+----6
             15.27$    16.3 DM             13 IS
```

Example 4: Mixed Justification

Issue the command TABS 10 20 30 40 50. Then issue the TABS line command. This displays the current tab positions as follows:

```
=tabs            *        *        *        *        *        *
```

Type an **L**, **R** or **D** next to each tab position as required (unmarked tab positions assume the value of the last **TABS** command):

```
=tabs            *R       *D       *D       *D       *L       *
```

After you press **Enter**, the input text line:

```
000010 %start    %0.01     %0.02     %0.03      %end
```

is displayed as follows:

```
=cols> ----+----1----+----2----+----3----+----4----+----5----+----6
            start        0.01      0.02      0.03       end
```

Example 5: Multiple Logical Tab Characters

Overtype the asterisks in the **=tabs** line with other special characters and specify left justification for each one as follows:

```
=tabs            ]L       &L       #L       $L       =L
```

After you press **Enter**, the input text line:

```
000010 =first$second#third&fourth]fifth
```

is displayed as follows:


```
  =cols> ----+----1----+----2----+----3----+----4----+----5----+----6
                                                          first
                                                  second
                                          third
                                  fourth
                          fifth
```

Example 6: Using a Blank as Logical Tab Character:

Issue the command:

```
TABS ' '
```

which activates tabulation with one blank as logical tab character. This means that words separated by one blank are tabulated. After you press **Enter**, the input text line:

```
000010 this is a blank tabulation
```

is displayed as follows:

```
=cols> ----+----1----+----2----+----3----+----4----+----5----+----6
                  this      is        a         blank     tabulation
```

